

A 2.542-Approximation for Precedence Constrained Single Machine Scheduling with Release Dates and Total Weighted Completion Time Objective

Martin Skutella*

April 25, 2016

Abstract

We present a $\sqrt{e}/(\sqrt{e}-1)$ -approximation algorithm for the nonpreemptive scheduling problem to minimize the total weighted completion time of jobs on a single machine subject to release dates and precedence constraints. The previously best known approximation algorithm dates back to 1997; its performance guarantee can be made arbitrarily close to the Euler constant e [18].

1 Introduction

We consider the following classical machine scheduling problem denoted by $1|r_j, prec|\sum w_j C_j$ in the standard classification scheme of Graham, Lawler, Lenstra, and Rinnooy Kan [12]. We are given a set of jobs $N = \{1, 2, \dots, n\}$ and for every job $j \in N$ a processing time $p_j \geq 0$, a release date $r_j \geq 0$, and a weight $w_j \geq 0$. The jobs $j \in N$ need to be processed during non-overlapping time intervals of length p_j , and j 's processing must not start before its release date r_j . Moreover, there are precedence constraints given by a partial order " \prec " on N where $j \prec k$ means that job j must be completed before job k may be started, that is, j 's processing interval must precede k 's. We may therefore without loss of generality assume throughout the paper that $j \prec k$ implies $r_j \leq r_k$. The objective is to minimize the total weighted completion time $\sum_{j \in N} w_j C_j$ where C_j denotes the first point in time at which j 's processing is completed.

*TU Berlin, Institut für Mathematik, MA 5-2, Straße des 17. Juni 136, 10623 Berlin, Germany, martin.skutella@tu-berlin.de

Complexity. Even for unit job weights, the special cases of the problem without non-trivial release dates $1|prec|\sum C_j$ (i.e., $r_j = 0$ for all $j \in N$) or without precedence constraints $1|r_j|\sum C_j$ are strongly NP-hard; see, e.g., [8, problem SS4]. In preemptive scheduling, the processing of a job may be repeatedly interrupted and resumed at a later point in time. In the absence of precedence constraints, the problem with unit job weights $1|r_j, pmtn|\sum C_j$ can be solved in polynomial time [3], but for arbitrary weights $1|r_j, pmtn|\sum w_j C_j$ is strongly NP-hard. Without non-trivial release dates preemptions are superfluous such that $1|prec, pmtn|\sum C_j$ is equivalent to $1|prec|\sum C_j$ and thus strongly NP-hard.

List scheduling. Before dipping into the rich history of approximation algorithms for these scheduling problems, we first discuss the most important algorithmic ingredient for both heuristic and exact solutions: *list scheduling*. Consider a list representing a total order on the set of jobs N , extending the given partial order “ \prec ”. A straightforward way to construct a feasible schedule is to process the jobs in the given order as early as possible with respect to release dates. A schedule constructed in this way is a *list schedule*.

Depending on the given list and the release dates of jobs, the machine might remain idle when one job is completed but the next job in the list is not yet released. On the other hand, if job preemptions are allowed, it is certainly not advisable to leave the machine idle while another job at a later position in the list is already available (released) and waiting. Instead, we better start this job and preempt it from the machine as soon as the next job in the list is released. In *preemptive list scheduling* we process at any point in time the first available job in the list. The resulting preemptive schedule is feasible (as $j \prec k$ implies $r_j \leq r_k$) and is called *preemptive list schedule*.

Known techniques and results. There is a vast literature on approximation algorithms for the various scheduling problems mentioned above. Here we only mention those results that are particularly relevant in the context of this paper and refer to Chekuri and Khanna [5] for a more comprehensive overview. Various kinds of linear programming (LP) relaxations have proved to be useful in designing approximation algorithms. One of the simplest and most intuitive classes of LP relaxations is based on completion time variables only. These LP relaxations were introduced by Queyranne [16] and first used in the context of approximation algorithms by Schulz [17], who presents a 2-approximation algorithm for the problem

$1|prec|\sum w_j C_j$ and a 3-approximation algorithm for $1|r_j, prec|\sum w_j C_j$; see also Hall, Schulz, Shmoys, and Wein [13]. These algorithms compute an optimal LP solution and then do list scheduling in order of increasing LP completion times. Moreover, Hall et al. [13] show that preemptive list scheduling in order of increasing LP completion times is a 2-approximation algorithm for $1|r_j, prec, pmtn|\sum w_j C_j$.

Phillips, Stein, and Wein [15] and Hall, Shmoys, and Wein [14] introduce the idea of list scheduling in order of so-called α -points to convert preemptive schedules to nonpreemptive ones. For $\alpha \in (0, 1]$, the α -point of a job with respect to a preemptive schedule is the first point in time when an α -fraction of the job has been completed. Goemans [10] and Chekuri, Motwani, Natarajan, and Stein [6] show that choosing α randomly leads to better results. In particular, Chekuri et al. [6] present an $e/(e-1)$ -approximation algorithm for $1|r_j|\sum C_j$ by starting from an optimal preemptive schedule. Goemans [10] and Goemans, Queyranne, Schulz, Skutella, and Wang [11] give approximation results for the more general weighted problem $1|r_j|\sum w_j C_j$ based on a preemptive schedule that is an optimal solution to an LP relaxation in time-indexed variables. Similarly, Schulz and Skutella [18] give an $(e + \varepsilon)$ -approximation algorithm for $1|r_j, prec|\sum w_j C_j$ for any $\varepsilon > 0$.

Bansal and Khot prove in a recent landmark paper [4] that there is no $(2 - \varepsilon)$ -approximation algorithm for $1|prec|\sum w_j C_j$, assuming a stronger version of the Unique Games Conjecture. Ambühl, Mastrolilli, Mutsanas, and Svensson [2], based on earlier work of Correa and Schulz [7] and Ambühl and Mastrolilli [1], prove an interesting relation between the approximability of $1|prec|\sum w_j C_j$ and the vertex cover problem

Our contribution. We present a $\sqrt{e}/(\sqrt{e} - 1)$ -approximation algorithm for $1|r_j, prec|\sum w_j C_j$ based on the following two ingredients: (i) For the problem $1|r_j, prec, pmtn|\sum w_j C_j$ we slightly strengthen the 2-approximation result of Hall et al. [13] and show that preemptive list scheduling in order of increasing LP completion times on a machine running at double speed yields a schedule whose cost is at most the cost of an optimal schedule on a regular machine; see Section 2. (ii) Modifying the analysis of Chekuri et al. [6] we show how to turn the preemptive schedule on the double speed machine into a nonpreemptive schedule on a regular machine while increasing the objective function by at most a factor of $\sqrt{e}/(\sqrt{e} - 1)$; see Section 3. We conclude with a conjecture in Section 4.

2 Optimal preemptive schedules under resource augmentation

In this section we consider the preemptive single machine scheduling problem with release dates, precedence constraints and total weighted completion time objective $1|r_j, prec, pmtn|\sum w_j C_j$. The best known approximation result for this problem is a 2-approximation algorithm due to Hall et al. [13] that is based on an LP relaxation in completion time variables originally introduced by Queyranne [16] and later refined by Goemans [9, 10] for problems involving release dates. Let $S \subseteq N$ denote a set of jobs and define

$$p(S) := \sum_{j \in S} p_j \quad \text{and} \quad r_{\min}(S) := \min_{j \in S} r_j.$$

The LP relaxation in completion time variables $C_j, j \in N$, looks as follows:

$$\begin{aligned} \min \quad & \sum_{j \in N} w_j C_j \\ \text{s.t.} \quad & C_j \leq C_k \quad \text{for all } j \prec k, \end{aligned} \quad (1)$$

$$\frac{1}{p(S)} \sum_{j \in S} p_j C_j \geq r_{\min}(S) + \frac{1}{2} p(S) \quad \text{for all } S \subseteq N. \quad (2)$$

Notice that constraints (1) could be strengthened to $C_j + p_k \leq C_k$, which is however not necessary for our purposes. Goemans [10] argues that constraints (2) hold for a feasible schedule, even if $(C_j)_{j \in N}$ denotes the vector of *mean busy times* of jobs instead of the larger completion times. Moreover, despite their exponential number, these constraints can be separated in polynomial time by efficient submodular function minimization [9]. Thus, an optimal solution C^* to the LP relaxation can be found in polynomial time and yields the LP lower bound $\sum_{j \in N} w_j C_j^*$ on the total weighted completion time of an optimal preemptive schedule. Reindex the set of jobs such that

$$C_1^* \leq C_2^* \leq \dots \leq C_n^* \quad \text{and} \quad (j \prec k \Rightarrow j < k). \quad (3)$$

The second condition in (3) is necessary to ensure that the total order of jobs by increasing indices extends the partial order given by the precedence constraints; notice, that in an optimal LP solution C_j^* might be equal to C_k^* for some pair of jobs with $j \prec k$.

Hall et al. [13] show that preemptive list scheduling according to list (3) yields a feasible preemptive schedule with completion times $C_j \leq 2 \cdot C_j^*$,

$j \in N$, and thus a 2-approximate solution. Exactly the same analysis implies a slightly stronger result in terms of resource augmentation as we show in the next lemma. We imagine a machine running at double speed such that each job $j \in N$ needs to be processed for $p_j/2$ time units only.

Lemma 1. *Preemptive list scheduling according to list (3) on a machine running at double speed yields a feasible preemptive schedule with completion times $C'_j \leq C_j^*$ for all $j \in N$.*

Proof. For a fixed $j \in N$, let S denote the subset of jobs $k \leq j$ such that (i) $C'_k \leq C'_j$, (ii) the preemptive list schedule does not leave the double speed machine idle between times C'_k and C'_j , and (iii) only jobs $\ell \leq j$ are being processed between times C'_k and C'_j .

Claim: The preemptive list schedule processes the set of jobs S without interruption during the time interval $I := [r_{\min}(S), C'_j]$.

To prove the claim, consider a job $h \in S$ with $r_h = r_{\min}(S)$. Since $\ell \prec h$ implies $r_\ell \leq r_h$, there is no idle time within the time interval $[r_h, C'_h]$ and only jobs $\ell \leq h \leq j$ are being processed there. Moreover, due to (ii) there is no idle time within the time interval $[C'_h, C'_j]$ and only jobs $\ell \leq j$ are being processed there due to (iii). As a consequence, there is no idle time in I and only jobs $\ell \leq j$ are being processed there. Therefore, every job k with $C'_k \in I$ satisfies conditions (i), (ii), and (iii), and is thus contained in S . Finally, since any job ℓ that the preemptive list schedule processes in I satisfies $\ell \leq j$ and is therefore completed before job j in I , the claim follows.

The claim implies that $C'_j = r_{\min}(S) + \frac{1}{2}p(S)$. Finally,

$$C_j^* \geq \frac{1}{p(S)} \sum_{k \in S} p_k C_k^* \geq r_{\min}(S) + \frac{1}{2}p(S) = C'_j,$$

where the first inequality holds as $C_k^* \leq C_j^*$ for $k \leq j$ and the second inequality follows by the LP constraints (2). \square

With the help of Lemma 1 we can now prove the main result of this section.

Theorem 1. *For a single machine running at double speed one can obtain in polynomial time a preemptive list schedule whose total weighted completion time is at most the LP lower bound $\sum_{j \in N} w_j C_j^*$ on the optimal total weighted completion time of a preemptive schedule for a regular single machine.*

Proof. As discussed above, an optimal solution C^* to the LP relaxation can be obtained in polynomial time. Our algorithm then applies preemptive list scheduling according to list (3) on a double speed machine. By Lemma 1, the total weighted completion time of the resulting preemptive list schedule is bounded from above by the LP lower bound $\sum_{j \in N} w_j C_j^*$. \square

3 Scheduling in order of alpha-points

In this section we show how to turn a preemptive schedule on the double speed machine into a nonpreemptive schedule on a regular machine while increasing the total weighted completion time by a factor at most 2.542.

Theorem 2. *Given a feasible preemptive list schedule S' on a double speed machine with completion times C'_j , $j \in N$, one can obtain in polynomial time a feasible nonpreemptive schedule on a regular speed machine with total weighted completion time*

$$\sum_{j \in N} w_j C_j \leq \frac{\sqrt{e}}{\sqrt{e} - 1} \sum_{j \in N} w_j C'_j.$$

Theorems 1 and 2 together yield the new approximation result for the scheduling problem under consideration.

The proof of Theorem 2 relies on list scheduling in order of α -points: For $0 < \alpha \leq 1$, the α -point $C'_j(\alpha)$ of job j with respect to schedule S' is the first point in time when job j has been processed for $\alpha \cdot p_j/2$ time on the double speed machine. Consider the list schedule S_α obtained by scheduling jobs in order of increasing $C'_j(\alpha)$ on a regular speed machine (notice that this order is in line with the precedence constraints as the preemptive schedule S' is feasible). Let C_j^α denote job j 's completion time in the list schedule S_α . Moreover, for a fixed job k , let η_j denote the fraction of job $j \in N$ that has been processed in schedule S' on the double speed machine by time C'_k . In particular,

$$C'_k \geq \sum_{j \in N} \eta_j \frac{p_j}{2}. \quad (4)$$

The following lemma is a slight modifications of a more general observations presented in [19, Corollary 2.3.3] and [20, Corollary 3.3]. We give a new, somewhat simpler proof.

Lemma 2.

$$C_k^\alpha \leq C'_k + \sum_{j: \eta_j \geq \alpha} \left(1 + \frac{\alpha - \eta_j}{2}\right) p_j. \quad (5)$$

Proof. Let X denote the subset of all jobs j scheduled by the list schedule S_α no later than job k such that there is no idle time between times C_j^α and C_k^α . In particular, $k \in X$. Moreover, let $\ell \in X$ denote the job that S_α schedules first among the jobs in X . By definition of X , job ℓ is started at its release date r_ℓ . In particular,

$$C_k^\alpha = r_\ell + \sum_{j \in X} p_j. \quad (6)$$

Since schedule S' obeys release dates, we get $r_\ell \leq C'_\ell(\alpha) \leq C'_j(\alpha)$ for every job $j \in X$. Moreover, by definition of η_j , schedule S' processes every job $j \in X$ in the time interval $[C'_j(\alpha), C'_k]$ for exactly $\frac{1}{2}(\eta_j - \alpha)p_j$ time units on the double speed machine. Since the machine can process at most one job at a time,

$$C'_k \geq r_\ell + \sum_{j \in X} \frac{\eta_j - \alpha}{2} p_j. \quad (7)$$

Combining (6) and (7) yields

$$C_k^\alpha \leq C'_k + \sum_{j \in X} \left(1 + \frac{\alpha - \eta_j}{2}\right) p_j. \quad (8)$$

Notice that the sum on the right hand side of (5) goes over a superset of X since $C'_j(\alpha) \leq C'_k(\alpha) \leq C'_k$ and thus $\eta_j \geq \alpha$ for every job $j \in X$. Finally, as $\alpha > 0$ and $\eta_j \leq 1$, the summand of every job j in (5) is at least $p_j/2$ and thus nonnegative. Therefore inequality (5) follows immediately from (8). \square

We now draw α randomly from $(0, 1]$ with density function

$$f(\alpha) := \frac{e^{\alpha/2}}{2(\sqrt{e} - 1)}.$$

Notice that for $0 \leq \eta \leq 1$

$$\int_0^\eta f(\alpha) \left(1 + \frac{\alpha - \eta}{2}\right) d\alpha = \frac{\eta}{2(\sqrt{e} - 1)}.$$

Thus, by Lemma 2,

$$\begin{aligned} \mathbb{E}[C_k^\alpha] &\leq C'_k + \sum_{j \in N} p_j \int_0^{\eta_j} f(\alpha) \left(1 + \frac{\alpha - \eta_j}{2}\right) d\alpha \\ &= C'_k + \frac{1}{\sqrt{e} - 1} \sum_{j \in N} \eta_j \frac{p_j}{2} \leq \frac{\sqrt{e}}{\sqrt{e} - 1} C'_k, \end{aligned}$$

where the last inequality follows from (4). By linearity of expectation, the expected total weighted completion time of the nonpreemptive list schedule in order of random α -points is at most $\sqrt{e}/(\sqrt{e} - 1)$ times larger than the total weighted completion time of the given preemptive schedule S' .

To complete the proof of Theorem 2 we need to derandomize the choice of α . In a preemptive list schedule, preemption of a job can only occur when another job is released. In particular, there can be at most $n - 1$ preemptions and therefore at most n combinatorially different choices of α . As observed by Goemans [10], an α minimizing the total weighted completion time of the resulting list schedule can thus be found in $O(n^2)$ time.

As a consequence of Theorems 1 and 2 we can state the following Corollary on the quality of the lower bound given by an optimal solution to the LP relaxation in Section 2.

Corollary 1. *For instances of the scheduling problem $1|r_j, prec|\sum w_j C_j$, the value of an optimal solution to the LP relaxation in completion time variables considered in Section 2 is at most a factor $(\sqrt{e} - 1)/\sqrt{e}$ smaller than the total weighted completion time of an optimal schedule.*

Proof. By Theorems 2 and 1 our approximation algorithm constructs a feasible schedule whose total weighted completion time is bounded by

$$\sum_{j \in N} w_j C_j \leq \frac{\sqrt{e}}{\sqrt{e} - 1} \sum_{j \in N} w_j C'_j \leq \frac{\sqrt{e}}{\sqrt{e} - 1} \sum_{j \in N} w_j C_j^*, \quad (9)$$

where C' denotes the vector of completion times of the preemptive double speed machine schedule S' and C^* is an optimal LP solution. Since the left-hand side of (9) is an upper bound on the total weighted completion time of an optimal schedule, the result follows. \square

4 Concluding remarks

Despite our enthusiastic yet ultimately fruitless efforts to improve the presented approximation result, we feel that the new performance guarantee

$\sqrt{e}/(\sqrt{e}-1)$ is hardly the last word on the considered scheduling problem. On the other hand, the history of approximation algorithms for the special case $1|prec|\sum w_j C_j$ and, in particular, recent non-approximability results make it seem somewhat unlikely to achieve a performance ratio strictly better than 2. Therefore, and due lack of imagination of other meaningful approximation ratios, we conclude with the following conjecture, granting an extra $+\varepsilon$ in the performance ratio to the release dates.

Conjecture 1. *For any $\varepsilon > 0$, there is a $(2 + \varepsilon)$ -approximation algorithm for $1|r_j, prec|\sum w_j C_j$.*

Acknowledgements. We are grateful to an anonymous referee for various helpful comments that have lead to an improved presentation of the paper. This work is supported by the Einstein Foundation Berlin.

References

- [1] C. Ambühl and M. Mastrolilli. Single machine precedence constrained scheduling is a vertex cover problem. *Algorithmica*, 53:488–503, 2009.
- [2] C. Ambühl, M. Mastrolilli, N. Mutsanas, and O. Svensson. On the approximability of single-machine scheduling with precedence constraints. *Mathematics of Operations Research*, 36:653–669, 2011.
- [3] K. R. Baker. *Introduction to Sequencing and Scheduling*. John Wiley & Sons, New York, 1974.
- [4] N. Bansal and S. Khot. Optimal long code test with one free bit. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 453–462, 2009.
- [5] C. Chekuri and S. Khanna. Approximation algorithms for minimizing average weighted completion time. In J. Leung, editor, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*. CRC Press, 2004.
- [6] C. S. Chekuri, R. Motwani, B. Natarajan, and C. Stein. Approximation techniques for average completion time scheduling. *SIAM Journal on Computing*, 31:146–166, 2001.
- [7] J. R. Correa and A. S. Schulz. Single machine scheduling with precedence constraint. *Mathematics of Operations Research*, 30:1005–1021, 2005.
- [8] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, 1979.
- [9] M. X. Goemans. A supermodular relaxation for scheduling with release dates. In W. H. Cunningham, S. T. McCormick, and M. Queyranne, editors, *Integer Programming and Combinatorial Optimization*, volume 1084 of *Lecture Notes in Computer Science*, pages 288–300. Springer, 1996.

- [10] M. X. Goemans. Improved approximation algorithms for scheduling with release dates. In *Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 591–598, New Orleans, LA, 1997.
- [11] M. X. Goemans, M. Queyranne, A. S. Schulz, M. Skutella, and Y. Wang. Single machine scheduling with release dates. *SIAM Journal on Discrete Mathematics*, 15:165–192, 2002.
- [12] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.
- [13] L. A. Hall, A. S. Schulz, D. B. Shmoys, and J. Wein. Scheduling to minimize average completion time: Off-line and on-line approximation algorithms. *Mathematics of Operations Research*, 22:513–544, 1997.
- [14] L. A. Hall, D. B. Shmoys, and J. Wein. Scheduling to minimize average completion time: Off-line and on-line algorithms. In *Proceedings of the 7th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 142–151, Atlanta, GA, 1996.
- [15] C. Phillips, C. Stein, and J. Wein. Minimizing average completion time in the presence of release dates. *Mathematical Programming*, 82:199–223, 1998.
- [16] M. Queyranne. Structure of a simple scheduling polyhedron. *Mathematical Programming*, 58:263–285, 1993.
- [17] A. S. Schulz. Scheduling to minimize total weighted completion time: Performance guarantees of LP-based heuristics and lower bounds. In W. H. Cunningham, S. T. McCormick, and M. Queyranne, editors, *Integer Programming and Combinatorial Optimization*, volume 1084 of *Lecture Notes in Computer Science*, pages 301–315. Springer, 1996.
- [18] A. S. Schulz and M. Skutella. Random-based scheduling: New approximations and LP lower bounds. In J. Rolim, editor, *Randomization and Approximation Techniques in Computer Science*, volume 1269 of *Lecture Notes in Computer Science*, pages 119–133. Springer, 1997.
- [19] M. Skutella. *Approximation and Randomization in Scheduling*. PhD thesis, Technische Universität Berlin, Germany, 1998.
- [20] M. Skutella. List scheduling in order of α -points on a single machine. In E. Bampis, K. Jansen, and C. Kenyon, editors, *Efficient Approximation and Online Algorithms: Recent Progress on Classical Combinatorial Optimization Problems and New Applications*, volume 3484 of *Lecture Notes in Computer Science*, pages 250–291. Springer, 2006.